

Amendments to the Claims

1 Claim 1 (currently amended): A method for programmatically enforcing referential integrity
2 constraints among associations between class instances, comprising steps of:
3 programmatically determining, when evaluating a request to modify an existing
4 association end of a bi-directional link to reflect an association from an instance of a first class to
5 an instance of a second class, whether the association end to be modified has a single multiplicity
6 or a many multiplicity;
7 if the association end to be modified has the single multiplicity, modifying the existing
8 association end by atomically and programmatically performing the steps of:
9 first disconnecting a previously-existing inverse association end of the requested
10 association end;
11 next setting an inverse association end of the association to reflect an inverse
12 association from the instance of the second class to the instance of the first class; and
13 then setting the requested association end from the instance of the first class to the
14 instance of the second class; and
15 if the association end to be modified has the many multiplicity, modifying the existing
16 association end by atomically and programmatically performing the steps of:
17 first adding the requested association end from the instance of the second first
18 class to the instance of the [[first]] second class;
19 next disconnecting the previously-existing inverse association end of the
20 requested association end; and
21 then setting an inverse association end of the association to reflect an inverse

Serial No. 09/827,290

-2-

Docket RSW920000173US1

22 association from the instance of the ~~[[first]]~~ second class to the instance of the ~~second~~ first class.

Claims 2 - 4 (canceled)

1 Claim 5 (previously presented): The method according to Claim 1, further comprising the step of
2 serializing the association by performing steps of:
3 determining whether the association end to be modified or the inverse association end is a
4 primary end of the association; and
5 serializing only the primary end of the association during the serialization.

1 Claim 6 (previously presented): The method according to Claim 1, wherein the method is
2 provided as a single link helper object and a multiple link helper object for each association,
3 wherein the single link helper object performs the atomically and programmatically performed
4 steps for the single multiplicity association end and the multiple link helper object performs the
5 atomically and programmatically performed steps for the many multiplicity association end.

1 Claim 7 (currently amended): A computer program product for programmatically enforcing
2 referential integrity constraints among associations between class instances, wherein the
3 computer program product is embodied on one or more computer readable media and comprises
4 computer-readable program code means for:
5 programmatically determining, when evaluating a request to modify an existing
6 association end of a bi-directional link to reflect an association from an instance of a first class to

Serial No. 09/827,290

-3-

Docket RSW920000173US1

an instance of a second class, whether the association end to be modified has a single multiplicity or a many multiplicity;

if the association end to be modified has the single multiplicity, modifying the existing association end by atomically and programmatically performing the steps of:

first disconnecting a previously-existing inverse association end of the requested association end;

next setting an inverse association end of the association to reflect an inverse association from the instance of the second class to the instance of the first class; and

then setting the requested association end from the instance of the first class to the instance of the second class; and

if the association end to be modified has the many multiplicity, modifying the existing association end by atomically and programmatically performing the steps of:

first adding the requested association end from the instance of the second first class to the instance of the [[first]] second class;

next disconnecting the previously-existing inverse association end of the requested association end; and

then setting an inverse association end of the association to reflect an inverse association from the instance of the [[first]] second class to the instance of the second first class~~[[.]]~~.

Claims 8 - 9 (canceled)

Serial No. 09/827,290

-4-

Docket RSW920000173US1

1 Claim 10 (previously presented): The computer program product according to Claim 7, further
2 comprising computer-readable program code means for serializing the association by performing
3 steps of:

4 determining whether the association end to be modified or the inverse association end is a
5 primary end of the association; and

6 computer-readable program code means for serializing only the primary end of the
7 association during the serialization.

1 Claim 11 (currently amended): A system for programmatically enforcing referential integrity
2 constraints among associations between class instances, comprising means for:

3 programmatically determining, when evaluating a request to modify an existing
4 association end of a bi-directional link to reflect an association from an instance of a first class to
5 an instance of a second class, whether the association end to be modified has a single multiplicity
6 or a many multiplicity;

7 if the association end to be modified has the single multiplicity, modifying the existing
8 association end by atomically and programmatically performing the steps of:

9 first disconnecting a previously-existing inverse association end of the requested
10 association end;

11 next setting an inverse association end of the association to reflect an inverse
12 association from the instance of the second class to the instance of the first class; and

13 then setting the requested association end from the instance of the first class to the
14 instance of the second class; and

15 if the association end to be modified has the many multiplicity, modifying the existing
16 association end by atomically and programmatically performing the steps of:
17 first adding the requested association end from the instance of the ~~second~~ first
18 class to the instance of the ~~[[first]]~~ second class;
19 next disconnecting the previously-existing inverse association end of the
20 requested association end; and
21 then setting an inverse association end of the association to reflect an inverse
22 association from the instance of the ~~[[first]]~~ second class to the instance of the ~~second~~ first class.

Claims 12 - 13 (canceled)

1 Claim 14 (previously presented): The system according to Claim 11, further comprising means
2 for serializing the association by performing steps of:
3 determining whether the association end to be modified or the inverse association end is a
4 primary end of the association; and
5 means for serializing only the primary end of the association during the serialization.

1 Claim 15 (previously presented): The method according to Claim 1, wherein one or more
2 structured markup language representations specify instances of the first class, instances of the
3 second class, and associations between the instances of the first and second classes.

1 Claim 16 (previously presented): The method according to Claim 15, wherein only one

Serial No. 09/827,290

-6-

Docket RSW920000173US1

2 association end for each association between instances is specified in the structured markup
3 language representations.

1 Claim 17 (previously presented): The method according to Claim 16, wherein the only one
2 association end is an association end designated as a primary end for the association.

1 Claim 18 (previously presented): The method according to Claim 15, wherein a serialization of
2 results of the request to modify the existing association end that has the single multiplicity
3 comprises the step of:

4 determining whether the association end to be modified is a primary end for the
5 association, and if so, programmatically performing the steps of:

6 removing the representation of the previously-existing inverse association end
7 from the structured markup language representation in which it is specified; and

8 adding a structured markup language representation of the new inverse association
9 end.